Vectorisation des processus d'appariement document-requête

Vincent Claveau* — **Romain Tavenard**** — **Laurent Amsaleg***

** IRISA-CNRS ** IRISA-ENS Cachan*
Campus de Beaulieu
F-35042 Rennes cedex
{Vincent.Claveau,Romain.Tavenard,Laurent.Amsaleg}@irisa.fr

RÉSUMÉ. *Dans la plupart des applications de RI, calculer rapidement la proximité entre documents et requêtes est crucial. Avec les modèles vectoriels, ce calcul se fait généralement de manière très efficace. Cependant, lorsque les requêtes sont très longues ou dans le cas de SRI basés sur des modèles plus avancés, ce calcul devient plus complexe et coûteux. Dans cet article, nous proposons une technique simple pour transformer n'importe quel processus d'appariement requête-document fournissant un score en un problème de calcul de distance entre vecteurs. Cette approche peut ainsi bénéficier des bonnes performances des outils existants d'indexation et de recherche approximative dans des espaces de grandes dimensions. Au travers de quelques expériences, nous montrons par ailleurs que cette représentation n'entraîne pas de baisse importante de qualité des résultats, et, lorsque de nombreux documents sont à retourner, améliore même le rappel par rapport au SRI original, à taille de résultat égal.*

ABSTRACT. *In most IR systems, rapidly computing the proximity between a query and a document is an issue. This is generally computed very efficiently in the Vector Space Model. When handling very long queries or with different IR models, however, the cost of this computation can be quite high. In this paper, we propose a simple approach transforming any document-query pairing technique into a vectorial representation. Therefore, it becomes possible to use existing approximate indexing techniques allowing the fast computation of distances between high-dimensional vectors. We experimentally show that our approach does not degrade the results and can even yields better recall rates when considering high document cut-off values.*

MOTS-CLÉS : *Vectorisation, modèle vectoriel, modèle de langue, complexité d'appariement*

KEYWORDS: *Vectorization, vector space, language modeling, pairing complexity*

1. Introduction

En recherche d'information, la rapidité de la recherche est un critère important pour de nombreuses applications. Si cette faible complexité temporelle est garantie pour certains modèles, ce n'est pas le cas pour des modèles de représentation complexes ou des techniques d'appariement document-requête sophistiquées. Dans de tels cas, la complexité du traitement d'une requête rapportée au nombre de documents exclut de traiter en-ligne des collections très grandes.

Dans cet article, nous proposons une approche dite de « vectorisation » répondant en partie à ces problèmes. Il s'agit de construire une représentation vectorielle des documents et de la requête à l'aide du système d'appariement initial. Chacune des composantes de ces vecteurs est le score de proximité obtenu, avec le système de RI original, entre le document (ou la requête) et un document-référence. Une fois ces vecteurs construits, répondre à une requête revient, comme dans le système vectoriel classique, à trouver les documents dont les vecteurs sont proches de celui de la requête. On transforme ainsi un appariement coûteux dans l'espace original du SRI en un appariement que l'on souhaite moins complexe dans un espace vectoriel euclidien de grande dimension. Cependant, ce calcul des distances entre ces nouveaux vecteurs, peu creux, peut se révéler cher s'il est fait de manière exhaustive et exacte. Des techniques de recherche approximative peuvent alors être utilisées pour réduire ce coût. L'intérêt en terme d'efficacité semble alors évident lorsque le processus d'appariement initial est très coûteux, puisqu'on ne le répète pas pour chaque document de la base, mais pour un nombre plus réduit de documents-références.

Dans les expériences présentées ici, le calcul des distances entre les vecteurs construits est exhaustif et exact. Cela permet évaluer les résultats optimaux que l'on peut attendre de cette approche sans la perte de précision liée aux calculs de distance approximatifs entre vecteurs. Nous ne nous intéressons donc pas au temps de calcul total de notre approche, mais seulement à la complexité de construction des vecteurs et à la qualité des résultats rendus. Celle-ci s'améliore d'ailleurs de manière inattendue : la comparaison avec les documents ne se faisant plus directement sur leur contenu, on trouve alors des documents pertinents ne contenant pas les termes de la requête.

La section suivante revient sur les différents aspects de cette vectorisation et leur traitement dans la littérature. Nous décrivons ensuite notre approche telle qu'elle est implémentée et testée (section 3). Expériences et résultats sont détaillés dans la section 4. Nous concluons par quelques perspectives.

2. Travaux connexes

Nous détaillons ici divers points de l'état de l'art liés à notre méthode de vectorisation. Nous revenons d'abord sur l'appariement document-requête dans différents modèles, discutons de leurs coûts puis d'autres techniques de RI exploitant des changements d'espace. Finalement, nous évoquons quelques techniques de recherche approximative calculant rapidement des distances dans les espaces vectoriels.

2.1. *Appariement document-requête*

Apparier documents et requêtes est une étape cruciale des SRI. Dans certains modèles, comme le modèle vectoriel classique, cet appariement est particulièrement efficace d'un point de vue calculatoire. Nous rappelons ci-dessous quelques-unes des conditions sur lesquelles repose cette efficacité. Nous montrons que si ces conditions ne sont plus assurées, cette efficacité est remise en question. Nous mettons ainsi en relief les différences entre notre système de vectorisation et le système vectoriel classique ou d'autres plus complexes dont nous évoquons quelques représentants ci-après.

Espaces vectoriels

Dans le modèle vectoriel (Salton *et al.,* 1975), documents et requêtes sont représentés par des vecteurs. L'appariement entre un document et une requête se fait en calculant la proximité de leurs vecteurs, le plus souvent en utilisant une distance de Minkowski, dite également de type Lp (V est l'ensemble des termes d'indexation de tous les documents) :

$$\delta_{Lp}(q,d) = \sqrt[p]{\sum_{t\in V}(q_t - d_t)^p}$$

Les valeurs les plus courantes sont $p = 1$ (distance L1, dite *manhattan* ou *city-block*), $p = 2$ (distance L2 ou euclidienne), ou $p \to \infty$ (distance de Chebyshev) ; p n'est pas nécessairement un entier mais doit être supérieur à 1 pour que soit respectée l'inégalité triangulaire. Rappelons que la distance basée sur le cosinus habituellement utilisée en RI textuelle est équivalente (i.e. produit le même ordonnancement des documents) à la distance L2 lorsque les vecteurs sont normalisés : $\delta_{L2}(q,d) = \sqrt{2 - 2 * \delta_{cos}(q,d)}$

Les mesures Lp, et donc le cosinus en particulier, ont l'avantage d'être rapides à calculer quand les vecteurs sont normés (sous Lp) et creux. En effet, on a, pour p entier :

$$\begin{aligned}
\delta_{Lp}(q,d) &= \sqrt[p]{\sum_{t\in V}\sum_{k=0}^{p} \mathsf{C}_p^k q_t^k * (-d_t)^{p-k}} \\
&= \sqrt[p]{\sum_{t\in V} q_t^p + (-1)^p \sum_{t\in V} d_t^p + \sum_{t\in V}\sum_{k=1}^{p-1} \mathsf{C}_p^k q_t^k * (-d_t)^{p-k}} \\
&= \sqrt[p]{1 + (-1)^p + \sum_{\substack{t \text{ tq } q_t\neq 0 \\ \text{and } d_t\neq 0}} \sum_{k=1}^{p-1} \mathsf{C}_p^k q_t^k * (-d_t)^{p-k}}
\end{aligned}$$

Il suffit donc d'examiner les composantes des vecteurs non nulles, à la fois pour la requête et le document. Dans le modèle vectoriel standard, cela revient à ne s'intéresser qu'aux mots partagés par la requête et le document, puisque les termes absents des documents (ou de la requête) ont une pondération nulle. Cela explique les mise en

œuvre par fichiers inversés et la grande rapidité de cette étape d'appariement, notamment lorsque l'on manipule des requêtes de quelques mots.

Cependant, parfois, ce processus d'appariement est beaucoup plus coûteux. C'est par exemple le cas lorsque les requêtes sont très longues (de la taille d'un document) ou lorsque les requêtes et/ou les documents sont enrichis par des connaissances lexicales (synonymes...) ou par *(pseudo-)relevance feedback*. De manière plus générale, c'est aussi le cas dès lors que le système de pondération adopté n'assigne pas 0 aux termes non présents dans les document et les requêtes. Enfin, le processus de recherche devient aussi très coûteux dès lors que la distance utilisée n'est plus une distance (ou une métrique) Lp, et que celle-ci prend en compte aussi bien les termes présents que les termes absents dans la requête et les documents. La complexité de cette recherche est alors fonction du nombre de documents dans la collection.

Modèles de langue

Dans les modèles de langue, apparier documents et requêtes se fait en calculant la probabilité que la requête soit engendrée par le modèle de langue de chaque document. La plupart du temps, des modèles unigrammes sont considérés. Suivant le modèle adopté pour décrire la distribution des mots dans les documents, les termes absents des requêtes peuvent être pris en compte. C'est le cas par exemple du modèle de Bernouilli multiple (Ponte *et al.,* 1998) ; le score d'un document d pour une requête q est dépend alors de la probabilité (M_d est le modèle du document) :

$$P(q|M_d) = \prod_{t_i \in q} P(t_i|M_d) * \prod_{t_i \notin q} (1 - P(t_i|M_d))$$

Par ailleurs, dans tous les modèles, pour éviter que l'absence d'un terme ne disqualifie complètement un document, les probabilités sont lissées. Cela conduit à ce que tous les termes du vocabulaire aient une probabilité non nulle dans tous les documents.

Ces éléments expliquent que l'examen exhaustif de la totalité de la collection est parfois nécessaire pour rechercher les documents pertinents. Cet examen peut s'avérer là encore très coûteux, notamment si les modèles sont plus sophistiqués (n-gram, modèles de langue syntaxiques...) et la collection grande.

Modèles flous

En RI floue, les documents sont considérés comme des ensembles flous : un terme appartient à un document à un certain degré (Pasi, 1999). Dans ces systèmes, il faut parfois considérer qu'un terme absent d'un document a un degré d'appartenance non-nul pour éviter les effets absorbant du 0 avec certains opérateurs flous d'appariement (Bosc *et al.,* 2009). L'examen de tous les documents de la collection peut donc être là aussi nécessaire et coûteux si le processus de calcul de score est complexe.

Autres

Beaucoup d'autres modèles de RI proposent des processus d'appariement complexes souvent très coûteux. C'est notamment le cas pour les techniques d'apparie-

ment de graphes (Dinh *et al.,* 2008, inter alia), et plus spécialement encore quand les métriques de comparaison de graphes s'appuient sur l'ensemble du graphe (Jeh *et al.,* 2002) est alors souvent prohibitif pour traiter de grosses collections de RI.

2.2. *Changement d'espace*

Plusieurs études ont proposé, pour différentes raisons, des représentations s'éloignant du modèle vectoriel standard. Parmi celles-ci, le *Generalized Vector Space Model* GVSM (Carbonell *et al.,* 1997), répondant aux critiques selon lesquelles les termes forment une mauvaise base pour l'espace vectoriel puisqu'ils ne sont pas indépendants les uns des autres. Le GVSM se place dans l'espace dual, où ce sont les documents, plus facilement considérés comme indépendants, qui forment les bases de l'espace.

Les techniques autour de LSI (Deerwester *et al.,* 1990) relèvent aussi de cette famille. Cela inclut les techniques exploitant pLSA (Hofmann, 1999), l'analyse en composante principale (ACP) (Berry *et al.,* 2005) et même des transformations linéaires aléatoires (Vempala, 2004). Ces méthodes ont toutes pour effet principal de réduire la matrice termes $\times$ documents du modèle vectoriel classique. Notre approche, appliquée à un système vectoriel, partage beaucoup de liens avec ces techniques. Mais elle est plus générique puisqu'elle s'applique à toute forme de modèle de RI, pourvu que celui-ci fournisse un score représentant la pertinence d'un document pour une requête.

2.3. *Recherche approximative*

Des techniques de calcul rapide des distances dans les espaces vectoriels ont également fait l'objet d'études. Ces techniques peuvent permettre à notre approche de diminuer très fortement sa complexité (cf. section 3.4). De manière générale, ces techniques troquent un fort gain en temps de réponse contre une légère perte de précision, les éléments retournés étant simplement similaires et non *les plus* similaires. Une approche est de découper l'espace des données en portions et de n'effectuer des recherches que sur une ou plusieurs portions (Stein, 2007, Datar *et al.,* 2004). Le NV-Tree (Lejsek *et al.,* 2008) suit cette approche, fabrique des portions à partir de l'enchaînement de multiples projections aléatoires des points de l'espace, n'analyse qu'une seule portion pour chaque requête, portion dont la taille est calculée pour ne générer qu'un unique accès disque. En complément, le NV-Tree calcule des distances approximatives, ce qui réduit encore le coût des appariements.

3. Vectorisation

Nous décrivons dans cette section les principes sous-tendant notre approche de vectorisation. Nous détaillons en particulier la façon dont sont constitués les documents-références servant de bases à l'espace vectoriel construit, ainsi que la complexité de cette approche.

3.1. *Principe*

L'idée au cœur de notre approche est de se servir d'un nombre m de documents dits *documents-références*. Pour chaque document de la collection, on calcule alors sa similarité, qui est un score, à chacune de ces références selon le modèle de similarité imposé par le SRI utilisé (chaque document de la collection joue le rôle de la requête). Les m valeurs obtenues sont rassemblées dans un vecteur qui caractérise désormais chaque document. Ce processus, éventuellement coûteux, s'effectue bien sûr hors-ligne. Les requêtes subissent le même traitement, en ligne, toutefois.

Les requêtes et les documents se trouvent donc représentés par des vecteurs de dimensions m. La distance entre une requête et les documents de la collection se calcule ensuite de manière similaire à ce qui se fait dans les modèles vectoriels, et ce indépendamment du modèle utilisé pour construire ces vecteurs.

Le recherche de documents pertinents pour une requête se fait donc de manière indirecte, par le bais des proximités avec les documents-références. La proximité finale peut donc être dite de second ordre. Cela implique aussi qu'un document puisse être jugé pertinent pour une requête même s'il ne contient aucun mot de cette requête. Bien entendu, pour que cette technique soit intéressante d'un point de vue calculatoire, il faut que $m < D$, où D est le nombre de documents de la collection.

3.2. *Constitution des documents-références*

De nombreuses alternatives existent pour constituer les m documents-références. Ceux-ci ne sont pas nécessairement issues de la collection, même s'il semble plus raisonnable que ce soit le cas ; on a ainsi une assurance plus grande d'une bonne adéquation, notamment du vocabulaire, entre ces références et les documents et requêtes à traiter. Dans les expériences présentées ci-après, les documents-références sont simplement des regroupements aléatoires de documents de la collection traitée qui forme une partition de l'ensemble des documents de départ. Les documents-références contiennent donc en moyenne D/m documents.

3.3. *Calcul de distances*

Notre processus de vectorisation permet de ramener n'importe quel modèle de RI à une représentation vectorielle. L'appariement requête-document se fait donc par un calcul de distance entre vecteurs, comme dans le modèle vectoriel classique. Les vecteurs sont normalisés (en cohérence avec la distance utilisée) ; dans les expériences présentées dans cet article, nous utilisons une distance L2. À la différence du modèle vectoriel classique où les vecteurs représentant les documents sont généralement extrêmement creux (et plus encore ceux des requêtes), les vecteurs obtenus par vectorisation n'ont pas forcément beaucoup de composantes à 0. Par conséquent, procéder à un calcul de distance exhaustif et exact est alors très coûteux, suggérant d'adopter plu-

tôt une stratégie approximative. Toutefois, dans les expériences reportées ci-dessous, nous avons pris le parti d'évaluer les résultats sans le biais d'approximations. Les calculs de distance se font donc de manière classique avec la distance L2.

3.4. *Complexité*

Le processus d'appariement requête-document mis en œuvre ici est donc composé d'une phase hors-ligne, possiblement coûteuse, et d'une phase en-ligne. Pour que notre approche soit efficace, cette phase en ligne doit être d'un coût calculatoire inférieur au coût de la phase d'appariement requête-document dans le modèle initial.

Ce coût varie selon le modèle de RI choisi (cf. section 2). Pour la plupart des modèles et sans perte de généralité, ce coût est $D * T_{score}$, où D est le nombre de documents dans la collection et T_{score} le temps de traitement pour un document résultant en un score. Pour la phase en-ligne de notre approche, ce même coût se décompose en un temps de calcul des scores sur les m documents-références, plus le temps de calcul des distances dans l'espace vectoriel construit, soit $m * T_{score} + T_{dist}$. Ce temps T_{dist}, s'il est fait avec une distance de Minkowski et de manière exhaustive et exacte comme dans les expériences que nous reportons ci-après, est lui-même de complexité $\mathcal{O}(D * m * c_{sparse})$ où c_{sparse} est le taux moyen d'éléments non vides dans les vecteurs. Cependant, comme nous l'avons vu en section 2.3, des techniques de recherche approximative comme le NV-Tree offrent de bien meilleures complexités : T_{dist} peut être ramené au temps d'un accès disque et est négligeable dans ce cas. Sous ces conditions, le gain en complexité peut donc aller jusqu'à un facteur $\frac{m}{D}$.

4. Expérimentations

4.1. *Données expérimentales*

Pour apprécier les différents effets de la vectorisation, nous utilisons deux collections aux caractéristiques très différentes, en français et provenant de la campagne d'évaluation Amaryllis. La première est la collection ELDA, petite collection de 3500 documents issus de questions/réponses de la commission européenne, accompagnée de 19 requêtes. La seconde est la collection INIST composée de 160 000 documents (résumés d'articles de diverses disciplines scientifiques) et de 30 requêtes.

Pour ces deux collections, les requêtes sont composées de plusieurs champs : titre, corps, description et concepts associés. Dans les expériences reportées ci-dessous, nous n'utilisons que le titre et le corps pour construire la requête effectivement envoyée aux systèmes. Aucun traitement particulier n'est effectué sur les documents.

4.2. *Modèles de représentation testés et calcul des distances*

Les effets de notre technique sont évalués sur deux modèles de RI : un modèle vectoriel classique et un modèle basé sur les modèle de langue. Le modèle vectoriel utilise les pondérations BM-25 (Okapi) avec les paramètres standard et la distance cosinus habituelle. Le système à base de modèle de langue est celui proposé par Hiemstra (Hiemstra, 1998). Dans ce modèle, pour éviter l'effet absorbant des probabilités nulles dans le calcul du score, les probabilités $P(t_i|M_d)$ sont calculées en fonction de leur présence dans le document examiné et dans l'ensemble de la collection.

Les expériences suivantes ont pour but de tester la validité de notre approche en terme de qualité des résultats. L'étape de calcul des distances entre vecteurs obtenus par vectorisation se fait de manière exacte et exhaustive. Les améliorations ou dégradations de performances constatées ne sont donc que le fait de la vectorisation et non des approximations dans le calcul des distances.

4.3. *Résultats*

Les résultats sont évalués par les mesures habituelles : rappel, précision à différents seuils, *mean average precision* (MAP), *11pt interpolated average precision* (IAP), R-prec. Un test de Wilcoxon ($p = 0.05$) est effectué pour s'assurer de la significativité des différences constatées ; lorsque c'est le cas, celles-ci sont notées en gras. Nous indiquons entre parenthèses les variations par rapport au modèle original.

Collection ELDA

Les tableaux 1 et 2 recensent les résultats obtenus à partir respectivement du système vectoriel et du modèle de langue. Dans les deux cas, nous indiquons les résultats du système original et ceux obtenus par vectorisation en utilisant 1750 et 3500 documents-références.

Dans ces résultats, on note tout d'abord que l'utilisation de notre technique améliore largement les résultats pourvu que les vecteurs construits soient de dimension suffisante. La figure 1, dans laquelle la MAP est relevée pour plusieurs valeurs de m, montre que ce seuil est atteint pour $m = 1500$ sur cette collection avec le modèle vectoriel. Cette amélioration n'apparaît pas dans le système basé sur les modèles de langue, où les résultats globaux sont comparables (pas de différences statistiquement significatives), mais où la précision est plus faible sur les 10 premiers documents et meilleure après. Cela peut s'expliquer par les très bonnes performances de l'approche originale qui donne de bien meilleurs résultats qu'Okapi et laisse donc moins de marge à l'amélioration.

Pour une petite collection comme celle-ci, utiliser 3500 documents-références, c'est-à-dire autant que le nombre de documents, mène à une dégradation des performances en temps de calcul. Il y a en effet autant de calculs de score pour construire les vecteurs qu'en utilisant le système original, puis il faut ensuite calculer en plus 3500

	BM-25 VSM	Vectorisation ($m = 1750$)	Vectorisation ($m = 3500$)
MAP	36.22	39.01 (+7.7 %)	**43.46 (+20 %)**
IAP	38.17	40.23 (+5.4 %)	**44.63 (+17 %)**
R-prec	37.47	40.00 (+6.8 %)	**44.29 (+18.2 %)**
P@5	54.00	55.33 (+2.5 %)	58.00 (+7.4 %)
P@10	47.67	**51.67 (+8.4 %)**	**54.67 (+14.7 %)**
P@50	30.00	29.07 (-3.1 %)	**33.53 (+11.8 %)**
P@100	20.73	19.87 (-4.2 %)	21.50 (+3.7 %)
P@500	5.99	5.71 (-4.8 %)	6.17 (+2.9 %)
P@1000	3.15	3.15 (0 %)	**3.27 (+3.9 %)**
P@3000	1.07	**1.17 (+9.0 %)**	**1.18 (+10 %)**

Tableau 1. *Performances d'un système basé sur un modèle vectoriel (BM-25) sur la collection ELDA (D=3500)*

	ML Hiemstra	Vectorisation ($m = 1750$)	Vectorisation ($m = 3500$)
MAP	53.50	48.01 (-10.3 %)	51.86 (-3.1 %)
IAP	54.19	48.97 (-9.6 %)	52.86 (-2.5 %)
R-prec	53.50	48.30 (-9.7 %)	52.09 (-2.6 %)
P@5	72.00	**61.33 (-14.8 %)**	**62.67 (-13 %)**
P@10	71.33	**58.33 (-18.2 %)**	**59.67 (-16.4 %)**
P@50	37.33	37.07 (-0.7 %)	**39.73 (+6.4 %)**
P@100	24.40	23.90 (-2.1 %)	**26.17 (+7.2 %)**
P@500	6.03	5.94 (-1.6 %)	**6.47 (+7.3 %)**
P@1000	3.17	3.21 (+1.3 %)	**3.38 (+6.6 %)**
P@3000	1.16	**1.18 (+1.4 %)**	**1.18 (+1.6 %)**

Tableau 2. *Performances d'un système basé sur un modèle de langue (Hiemstra) sur la collection ELDA (D=3500)*

distances sur les vecteurs construits par la vectorisation. Il n'y a donc pas d'avantage en terme d'efficacité à utiliser ce processus de vectorisation pour de toutes petites collections. En revanche, il y a un avantage en terme de résultats, surtout si on se place dans des problématiques dans lesquelles trouver la totalité des documents répondant aux besoins de recherche est important (eg. recherche d'antériorité pour les brevets). Un examen manuel des résultats montre comme attendu que certains documents retrouvés par vectorisation mais pas par le système initial ne comportent en fait aucun mot de la requête.

Il est intéressant de constater que certaines grandes variations (améliorations ou dégradations des résultats par rapport au système original) ne sont pas forcément statistiquement significatives. Cela indique une grande variabilité selon les requêtes et est confirmé par l'examen des résultats requête par requête.

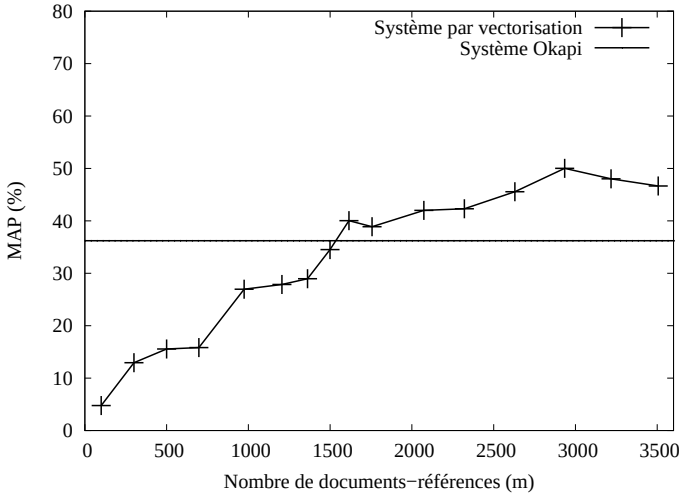

Figure 1. *Évolution de la MAP selon la dimension m des vecteurs, collection ELDA*

Collection INIST

Le tableau 3 recense les résultats obtenus sur la collection INIST pour une vectorisation avec 10 000 documents-références, pour le modèle vectoriel et le modèle de langue. Les performances obtenues sur les deux systèmes sont homogènes et confirment les observations faites sur la collection ELDA. On constate donc ici aussi des résultats globaux assez similaires entre la version originale et la vectorisation, avec une baisse de précision sur les premiers documents et une hausse de précision ensuite. Comme précédemment, la plus faible ampleur des améliorations apportées par la vectorisation sur les modèles de langues s'explique sans doute par les bonnes performances initiales du système de Hiemstra. Par ailleurs, là encore, un examen manuel des listes de réponses montre que certains documents pertinents trouvés uniquement par vectorisation ne contiennent aucun mot de la requête considérée.

Sur cette collection, l'intérêt de la technique en terme de complexité est en revanche bien plus évident : pour le modèle vectoriel comme pour le modèle de langue, le temps CPU de construction des 10 000 vecteurs est largement inférieur à celui de la recherche avec les modèles standard puisqu'il nécessite 16 fois moins de calcul. Comme nous l'avons expliqué, cet avantage est gommé par le calcul de proximité nécessaire à la vectorisation, puisque ce dernier est fait de manière exacte et exhaustive dans les expériences présentées ici. Ces résultats laissent néanmoins espérer d'excellentes performances avec l'utilisation d'outils tels que le NV-Tree.

	BM-25 VSM	Vectorisation	ML Hiemstra	Vectorisation
MAP	14.52	14.26 (-1.8 %)	16.94	16.20 (-4.4 %)
IAP	16.60	16.14 (-2.8 %)	19.38	19.13 (-1.3 %)
R-prec	18.77	**17.06 (-9.1 %)**	20.60	19.61 (-4.8 %)
P@5	41.33	**32.00 (-22.6 %)**	45.33	**37.74 (-16.7 %)**
P@10	34.00	**29.00 (-14.7 %)**	39.33	**34.04 (-13.5 %)**
P@50	18.47	18.00 (-2.5 %)	20.73	19.71 (-4.9 %)
P@100	12.00	**13.47 (+12.2 %)**	13.73	13.95 (+1.6 %)
P@500	4.27	**4.93 (+15.3 %)**	4.91	**5.37 (+9.4 %)**
P@1000	2.46	**2.89 (+17.3 %)**	2.82	**3.08 (+9.2 %)**
P@3000	1.01	**1.12 (+10.9 %)**	1.08	**1.16 (+7.4 %)**

Tableau 3. *Performances de BM-25 et de Hiemstra sur la collection INIST (D=160 000,* $m = 10\,000$*)*

5. Conclusions et perspectives

L'approche par vectorisation présentée dans cet article offre un grand intérêt en terme de complexité dans le cas de techniques d'appariement coûteuses et de grandes collections. Mais nous avons montré que même pour des techniques peu coûteuses, comme le modèle vectoriel, cette approche a un intérêt en terme de qualité de résultats, notamment pour des systèmes dans lesquels le rappel se doit d'être bon.

Ce travail ouvre beaucoup de perspectives. D'un point de vue théorique, il serait intéressant d'explorer les liens entre les approches LSI et la vectorisation lorsqu'elle est appliquée à un modèle vectoriel classique. De même, il nous semble indispensable d'analyser les liens qu'a notre approche avec celles de type *embedding* ((Bourgain, 1985, Abraham *et al.,* 2006)). D'un point de vue plus expérimental, nous étudions actuellement l'utilisation des techniques de recherche approximative décrites précédemment (LSH et le NV-Tree) en conjonction de notre approche. Cela devrait nous permettre obtenir des systèmes extrêmement rapides et à la performance contrôlée.

Nous projetons également l'étude exhaustive du comportement d'une telle technique sur de très grosses collections pour examiner la variation des performances selon la taille des vecteurs adoptée. Des tests sur de telles collections avec des systèmes d'appariement beaucoup plus coûteux que ceux décrits ici, comme dans les modèles à base de graphes, mettraient en valeur la faible complexité de notre approche.

Différentes stratégies pour constituer les documents-références sont également à l'étude. Parmi celles-ci, l'utilisation de techniques de *clustering* semble attractif. Cela permettrait d'obtenir des documents plus homogènes qui mèneraient à des dimensions plus indépendantes pour les vecteurs construits.

Enfin, cette approche par vectorisation permet aussi d'offrir une représentation unifiée à des systèmes de recherche différents. Il est ainsi possible de mélanger très simplement plusieurs systèmes hétérogènes pour construire les vecteurs représentatifs

des documents et des requêtes, chacun participant pour un certain nombre de dimension aux vecteurs résultants.

6. Bibliographie

Abraham I., Bartal Y., Neiman O., « Advances in metric embedding theory », *Proc. of Symposium on Theory Of Computing*, Seattle, USA, 2006.

Berry M., Martin D., « Principal Component Analysis for Information Retrieval », *in* , E. Kontoghiorghes (ed.), *Handbook of Parallel Computing and Statistics*, Statistics : A Series of Textbooks and Monographs, 2005.

Bosc P., Claveau V., Pivert O., Ughetto L., « Graded-Inclusion-Based Information retrieval Systems », *Proc. of European Conference on Information Retrieval*, LNCS 5478, Toulouse, France, 2009.

Bourgain J., « On lipschitz embedding of finite metric spaces in Hilbert space », *Israel Journal of Mathematics*, 1985.

Carbonell J. G., Yang Y., Frederking R. E., Brown R. D., Geng Y., Lee D., « Translingual Information Retrieval : A Comparative Evaluation », *Proc. of IJCAI*, Nagoya, Japon, 1997.

Datar M., Immorlica N., Indyk P., Mirrokni V., « Locality-Sensitive Hashing Scheme Based on p-Stable Distributions », *Proc. of the 20th ACM Symposium on Computational Geometry*, Brooklyn, New York, USA, 2004.

Deerwester S., Dumais S. T., Furnas G. W., Landauer T. K., Harshman R., « Indexing by Latent Semantic Analysis », *Journal of the American Society for Information Science*, 1990.

Dinh Q., Dkaki T., Mothe J., Charrel J.-P., « Information retrieval model based on graph comparison », *Actes des 9èmes Journée internationales d'Analyses statistiques de Données Textuelles, JADT'08*, Rome, Italie, 2008.

Hiemstra D., « A linguistically motivated probabilistic model of information retrieval », *Proc. of European Conference on Digital Libraries, ECDL*, Heraklion, Greece, 1998.

Hofmann T., « Probabilistic latent semantic indexing », *Proc. of SIGIR*, Berkeley, USA, 1999.

Jeh G., Widom J., « SimRank : a measure of structural-context similarity », *Proc. of SIGKDD*, Edmonton, Canada, 2002.

Lejsek H., Asmundsson F., Jónsson B., Amsaleg L., « NV-tree : An Efficient Disk-Based Index for Approximate Search in Very Large High-Dimensional Collections », *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2008.

Pasi G., « A logical formulation of the Boolean model and of weighted Boolean models », *LUMIS workshop at ECSQARU'99*, Londres, Grande-Bretagne, 1999.

Ponte J. M., Croft W. B., « A Language Modeling Approach to Information Retrieval », *Proc. of SIGIR*, Melbourne, Australie, 1998.

Salton G., Wong A., Yang C. S., « A Vector Space Model for Automatic Indexing », *Comm. of the ACM*, 1975.

Stein B., « Principles of hash-based text retrieval », *Proc. of SIGIR*, Amsterdam, Pays-Bas, 2007.

Vempala S., *The Random Projection Method*, vol. 65 of *Discrete Mathematics and Theoretical Computer Science*, AMS, 2004.